

Malware Analysis for Machine Learning

Generated by Doxygen 1.8.11

Contents

1	Malware Analysis for Machine Learning	1
2	Hierarchical Index	3
2.1	Class Hierarchy	3
3	Class Index	5
3.1	Class List	5
4	File Index	7
4.1	File List	7
5	Class Documentation	9
5.1	Class Class Reference	9
5.1.1	Detailed Description	10
5.1.2	Constructor & Destructor Documentation	10
5.1.2.1	Class(const std::string &name)	10
5.1.2.2	Class(const std::string &name, const std::set< int > &sets)	10
5.1.2.3	~Class()	11
5.1.3	Member Function Documentation	11
5.1.3.1	getLogProbability(const std::string &prefix, const std::string &value) const	11
5.1.3.2	getLogProbability() const	11
5.1.3.3	getSetId(const std::string &prefix) const	11
5.1.3.4	name() const	12
5.1.3.5	nbManifests() const	12
5.1.3.6	setNbValue()	12

5.1.3.7	setProbability(const uint total)	12
5.1.3.8	study(const boost::filesystem::path &filePath)	12
5.1.4	Friends And Related Function Documentation	12
5.1.4.1	operator<<	13
5.1.5	Member Data Documentation	13
5.1.5.1	m_name	13
5.1.5.2	m_nbMan	13
5.1.5.3	m_nbValue	13
5.1.5.4	m_pL	13
5.1.5.5	m_sets	13
5.2	FamilyClassifier Class Reference	13
5.2.1	Detailed Description	14
5.2.2	Constructor & Destructor Documentation	14
5.2.2.1	FamilyClassifier(const boost::filesystem::path &pDS, const boost::filesystem::path &pHD, const std::set< int > &sets, const uint percent)	14
5.2.2.2	~FamilyClassifier()	15
5.2.3	Member Function Documentation	15
5.2.3.1	evaluate(const uint nb, const uint to, const uint from) const	15
5.2.3.2	train()	15
5.3	MalwareDetector Class Reference	16
5.3.1	Detailed Description	17
5.3.2	Constructor & Destructor Documentation	17
5.3.2.1	MalwareDetector(const boost::filesystem::path &pDS, const boost::filesystem::path &pHD, const std::set< int > &sets, const uint percent)	17
5.3.2.2	~MalwareDetector()	17
5.3.3	Member Function Documentation	17
5.3.3.1	evaluate(const uint nb, const uint to, const uint from) const	17
5.3.3.2	train()	18
5.4	NaiveBayesClassifier Class Reference	18
5.4.1	Detailed Description	19
5.4.2	Constructor & Destructor Documentation	19

5.4.2.1	NaiveBayesClassifier(const boost::filesystem::path &pDS, const boost::filesystem::path &pHD, const std::set< int > &sets, const uint percent)	19
5.4.2.2	~NaiveBayesClassifier()	20
5.4.3	Member Function Documentation	20
5.4.3.1	evaluate(const uint nbTests) const	20
5.4.3.2	evaluate(const uint nb, const uint to, const uint from) const =0	20
5.4.3.3	getClass(const boost::filesystem::path &filePath) const	20
5.4.3.4	train()=0	21
5.4.4	Member Data Documentation	21
5.4.4.1	m_class	21
5.4.4.2	m_hashMalware	21
5.4.4.3	m_pathDataSet	21
5.4.4.4	m_pathHashDico	21
5.4.4.5	m_percent	21
5.4.4.6	m_sets	21
5.4.4.7	m_sizeTest	21
5.4.4.8	m_sizeTrain	22
5.4.4.9	m_total	22
5.5	SubSet Class Reference	22
5.5.1	Detailed Description	23
5.5.2	Constructor & Destructor Documentation	23
5.5.2.1	SubSet(const int id)	23
5.5.2.2	~SubSet()	23
5.5.3	Member Function Documentation	23
5.5.3.1	add(const std::string &value)	23
5.5.3.2	getId() const	24
5.5.3.3	getLogProbability(const std::string &value)	24
5.5.3.4	getOcc(const std::string &value)	24
5.5.3.5	getOcc() const	25
5.5.3.6	name() const	25
5.5.3.7	name(const uint id)	25

5.5.4	Friends And Related Function Documentation	25
5.5.4.1	operator<<	25
5.5.5	Member Data Documentation	25
5.5.5.1	m_id	25
5.5.5.2	m_nbLValue	26
5.5.5.3	m_nbValue	26
5.5.5.4	m_root	26
5.5.5.5	setNames	26
5.6	Value Class Reference	26
5.6.1	Detailed Description	27
5.6.2	Constructor & Destructor Documentation	28
5.6.2.1	Value()	28
5.6.2.2	Value(const std::string &value)	28
5.6.2.3	~Value()	28
5.6.3	Member Function Documentation	28
5.6.3.1	addNext(const std::string &value)	28
5.6.3.2	addOcc()	28
5.6.3.3	getOcc() const	28
5.6.3.4	getValue() const	29
5.6.3.5	goNext(const std::string &value)	29
5.6.3.6	isInNext(const std::string &value) const	29
5.6.3.7	print(std::ostream &out, const std::string offset) const	29
5.6.4	Friends And Related Function Documentation	30
5.6.4.1	operator<<	30
5.6.5	Member Data Documentation	30
5.6.5.1	m_next	30
5.6.5.2	m_occ	30
5.6.5.3	m_value	30

6 File Documentation	31
6.1 Classifiers.cpp File Reference	31
6.1.1 Detailed Description	31
6.2 Classifiers.hpp File Reference	32
6.2.1 Detailed Description	32
6.3 define.hpp File Reference	33
6.3.1 Detailed Description	34
6.3.2 Macro Definition Documentation	34
6.3.2.1 BAD_BOUND	34
6.3.2.2 DEBUG_ON	34
6.3.2.3 DEFAULT_C_NBTESTS	34
6.3.2.4 DEFAULT_C_PERCENT	34
6.3.2.5 DEFAULT_D_NBTESTS	34
6.3.2.6 DEFAULT_D_PERCENT	34
6.3.2.7 DEFAULT_MINNUMBERSAMPLE	34
6.3.2.8 DEFAULT_PATH_DATASET	34
6.3.2.9 DEFAULT_PATH_HASHDICO	34
6.3.2.10 DEFAULT_SETS	35
6.3.2.11 GOOD_BOUND	35
6.3.2.12 TOKENIZERS	35
6.3.3 Function Documentation	35
6.3.3.1 colorizedValue(const double value, const bool invert)	35
6.3.3.2 divide(const long double a, const long double b)	35
6.3.3.3 stringNULL("0")	35
6.3.3.4 toLower(const std::string &in)	35
6.4 Main.cpp File Reference	35
6.4.1 Detailed Description	35
6.4.2 Function Documentation	36
6.4.2.1 main(int argc, char *argv[])	36
6.5 README.md File Reference	36

6.6	Util.cpp File Reference	36
6.6.1	Detailed Description	36
6.6.2	Function Documentation	37
6.6.2.1	colorizedValue(const double value, const bool invert)	37
6.6.2.2	divide(const long double a, const long double b)	37
6.6.2.3	toLower(const string &in)	37
6.7	VectorSpace.cpp File Reference	37
6.7.1	Detailed Description	37
6.7.2	Function Documentation	37
6.7.2.1	operator<<(ostream &out, const Class &c)	37
6.7.2.2	operator<<(ostream &out, const SubSet &ss)	38
6.7.2.3	operator<<(ostream &out, const Value &v)	38
6.8	VectorSpace.hpp File Reference	38
6.8.1	Detailed Description	39
	Index	41

Chapter 1

Malware Analysis for Machine Learning

Getting Started

These instructions will get you a copy of the project up and running on your local machine for development and testing purposes.

Prerequisites

Requires boost libraries, boost filesystem module and a compiler supporting c++11 standards

```
1 sudo apt-get install libboost-all-dev
```

Installing

Create the build directory if not present at ROOT

```
1 cd ROOT/  
2 mkdir build
```

then

```
1 cd build  
2 cmake ..  
3 make  
4 cd ..
```

You can also run the following commands directly into the ROOT folder, but it'll flood it with unwanted files

```
1 cmake ./  
2 make
```

Running the programs

The two following classifiers are generate in the ROOT directory

I assume that the training set are located here too, otherwise, the default paths can be changed with options

malware detection

To use the **malware detector** with default arguments, run

```
1 cd ROOT/  
2 ./detector
```

For help and detailed options, run

```
1 ./detector help
```

malware classification

To use the **malware family classifier** with default arguments, run

```
1 cd ROOT/  
2 ./classifier
```

For help and detailed options, run

```
1 ./classifier help
```

Author

Aurelien BEC - *A Machine Learning homework* - [AurelBec](#)

License

This project is licensed under free licence

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Class	9
NaiveBayesClassifier	18
FamilyClassifier	13
MalwareDetector	16
SubSet	22
Value	26

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Class		
	Parent Classifier	9
FamilyClassifier		
	Malware family classifier (inherited from NaiveBayesClassifier)	13
MalwareDetector		
	Malware detector (inherited from NaiveBayesClassifier)	16
NaiveBayesClassifier		
	Parent Classifier	18
SubSet		
	Class SubSet	22
Value		
	Class Value	26

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Classifiers.cpp	Contains the implemented methods of the different classifiers	31
Classifiers.hpp	Contains the definition of different classifiers	32
define.hpp	Contains the global definitions for the project	33
Main.cpp	Contains the main function	35
Util.cpp	Contains some useful functions for the project	36
VectorSpace.cpp	Contains the implemented methods of VectorSpace and the other class need	37
VectorSpace.hpp	Contains the definition of VectorSpace and the other class need	38

Chapter 5

Class Documentation

5.1 Class Class Reference

Parent Classifier.

```
#include <VectorSpace.hpp>
```

Public Member Functions

- [Class](#) (const std::string &name)
Constructor.
- [Class](#) (const std::string &name, const std::set< int > &sets)
Constructor.
- [~Class](#) ()
Destructor.
- void [study](#) (const boost::filesystem::path &filePath)
Study a manifest.
- const std::string & [name](#) () const
Get the name of the [Class](#).
- const uint [nbManifests](#) () const
Get the size of the [Class](#).
- void [setNbValue](#) ()
Set the number of value in the [Class](#).
- void [setProbability](#) (const uint total)
Set the probability of occurrence of the [Class](#).
- const long double [getLogProbability](#) (const std::string &prefix, const std::string &value) const
Get the natural logarithmic probability of occurrence of a value.
- const long double [getLogProbability](#) () const
Get the probability of occurrence of the [Class](#).

Private Member Functions

- const int [getSetId](#) (const std::string &prefix) const
Get the id of a [SubSet](#).

Private Attributes

- uint [m_nbMan](#)
Number of manifests studied during training phase.
- uint [m_nbValue](#)
Number of value studied during training phase.
- long double [m_pL](#)
Natural logarithmic probability of [Class](#)' occurrence.
- std::vector< [SubSet](#) * > [m_sets](#)
Sets considered by the classifier.
- const std::string [m_name](#)
Name of the [Class](#).

Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [Class](#) &c)

5.1.1 Detailed Description

Parent Classifier.

This class is used to create a classifier

5.1.2 Constructor & Destructor Documentation

5.1.2.1 `Class::Class (const std::string & name) [inline]`

Constructor.

Constructor of the class [Class](#)

Parameters

<i>name</i>	The name of the Class
-------------	---------------------------------------

5.1.2.2 `Class::Class (const std::string & name, const std::set< int > & sets) [inline]`

Constructor.

Only constructor of the class [SubSet](#)

Parameters

<i>name</i>	The name of the Class
<i>sets</i>	The ids of the SubSet the Class have to considered

5.1.2.3 Class::~~Class () [inline]

Destructor.

Destructor of the class [Class](#)

5.1.3 Member Function Documentation

5.1.3.1 const long double Class::getLogProbability (const std::string & *prefix*, const std::string & *value*) const

Get the natural logarithmic probability of occurrence of a value.

Gets the probability of occurrence of a value in the [Class](#)

Parameters

<i>prefix</i>	The set in which look for the value
<i>value</i>	The value to look for

Returns

The natural logarithmic probability of occurrence of a value

5.1.3.2 const long double Class::getLogProbability () const [inline]

Get the probability of occurrence of the [Class](#).

Returns the probability of occurrence of the [Class](#)

Returns

The probability of occurrence of the [Class](#)

5.1.3.3 const int Class::getSetId (const std::string & *prefix*) const [private]

Get the id of a [SubSet](#).

Returns the id of a [SubSet](#)

Parameters

<i>prefix</i>	The name of the SubSet to look for
---------------	--

Returns

The id of the [SubSet](#)

5.1.3.4 `const std::string& Class::name () const` `[inline]`

Get the name of the [Class](#).

Returns the name of the [Class](#)

Returns

The name of the [Class](#)

5.1.3.5 `const uint Class::nbManifests () const` `[inline]`

Get the size of the [Class](#).

Returns the size of the [Class](#), the number of manifest studied

Returns

The size of the [Class](#)

5.1.3.6 `void Class::setNbValue ()` `[inline]`

Set the number of value in the [Class](#).

Sets the number of value studied in the [Class](#)

5.1.3.7 `void Class::setProbability (const uint total)` `[inline]`

Set the probability of occurrence of the [Class](#).

Sets the probability of the occurrence of the [Class](#)

5.1.3.8 `void Class::study (const boost::filesystem::path & filePath)`

Study a manifest.

Learns data from a manifest during the training phase

Parameters

<i>filePath</i>	The path to the manifest
-----------------	--------------------------

CLASS method's definitions

5.1.4 Friends And Related Function Documentation

5.1.4.1 `std::ostream& operator<< (std::ostream & out, const Class & c)` `[friend]`

5.1.5 Member Data Documentation

5.1.5.1 `const std::string Class::m_name` `[private]`

Name of the [Class](#).

5.1.5.2 `uint Class::m_nbMan` `[private]`

Number of manifests studied during training phase.

5.1.5.3 `uint Class::m_nbValue` `[private]`

Number of value studied during training phase.

5.1.5.4 `long double Class::m_pL` `[private]`

Natural logarithmic probability of [Class](#)' occurrence.

5.1.5.5 `std::vector<SubSet*> Class::m_sets` `[private]`

Sets considered by the classifier.

The documentation for this class was generated from the following files:

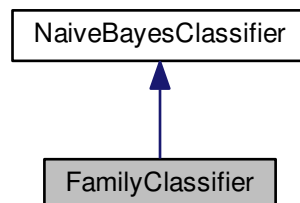
- [VectorSpace.hpp](#)
- [VectorSpace.cpp](#)

5.2 FamilyClassifier Class Reference

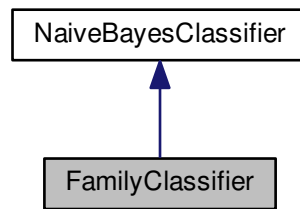
Malware family classifier (inherited from [NaiveBayesClassifier](#))

```
#include <Classifiers.hpp>
```

Inheritance diagram for FamilyClassifier:



Collaboration diagram for FamilyClassifier:



Public Member Functions

- [FamilyClassifier](#) (const boost::filesystem::path &pDS, const boost::filesystem::path &pHD, const std::set< int > &sets, const uint percent)
Constructor.
- [~FamilyClassifier](#) ()
Destructor.

Private Member Functions

- void [train](#) ()
Train.
- void [evaluate](#) (const uint nb, const uint to, const uint from) const
Evaluate.

Additional Inherited Members

5.2.1 Detailed Description

Malware family classifier (inherited from [NaiveBayesClassifier](#))

This class is used to create a malware family classifier

5.2.2 Constructor & Destructor Documentation

5.2.2.1 `FamilyClassifier::FamilyClassifier (const boost::filesystem::path & pDS, const boost::filesystem::path & pHD, const std::set< int > & sets, const uint percent)`

Constructor.

Constructor of the inherited class [FamilyClassifier](#)

Parameters

<i>pDS</i>	RELATIVE path to the directory containing all the examples
<i>pHD</i>	RELATIVE path to the SHA1 HASH file, containing name and family of known malware
<i>sets</i>	The ids of the sets considered by the classifier
<i>percent</i>	Percent of all available examples used for training

FAMILY CLASSIFIER method's definitions

5.2.2.2 FamilyClassifier::~FamilyClassifier () [inline]

Destructor.

Destructor of the inherited class [FamilyClassifier](#)

5.2.3 Member Function Documentation

5.2.3.1 void FamilyClassifier::evaluate (const uint *nb*, const uint *to*, const uint *from*) const [private],[virtual]

Evaluate.

Classifies *nb* examples in the domain selected and show results

Parameters

<i>nb</i>	Number of examples to classify
<i>to</i>	Position of the first file of the domain
<i>from</i>	Position of the last file of the domain

(0 <= *nb* <= *from* - *to*)

Implements [NaiveBayesClassifier](#).

5.2.3.2 void FamilyClassifier::train () [private],[virtual]

Train.

Train our classifier with *m_sizeTrain* examples

Implements [NaiveBayesClassifier](#).

The documentation for this class was generated from the following files:

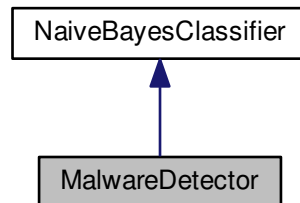
- [Classifiers.hpp](#)
- [Classifiers.cpp](#)

5.3 MalwareDetector Class Reference

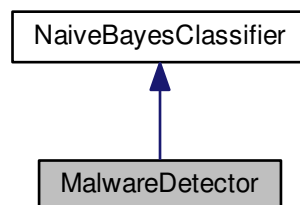
Malware detector (inherited from [NaiveBayesClassifier](#))

```
#include <Classifiers.hpp>
```

Inheritance diagram for MalwareDetector:



Collaboration diagram for MalwareDetector:



Public Member Functions

- [MalwareDetector](#) (const boost::filesystem::path &pDS, const boost::filesystem::path &pHD, const std::set<int> &sets, const uint percent)
Constructor.
- [~MalwareDetector](#) ()
Destructor.

Private Member Functions

- void [train](#) ()
Train.
- void [evaluate](#) (const uint nb, const uint to, const uint from) const
Evaluate.

Additional Inherited Members

5.3.1 Detailed Description

Malware detector (inherited from [NaiveBayesClassifier](#))

This class is used to create a malware detector

5.3.2 Constructor & Destructor Documentation

5.3.2.1 `MalwareDetector::MalwareDetector (const boost::filesystem::path & pDS, const boost::filesystem::path & pHD, const std::set< int > & sets, const uint percent)`

Constructor.

Constructor of the inherited class MalwareClassifier

Parameters

<i>pDS</i>	RELATIVE path to the directory containing all the examples
<i>pHD</i>	RELATIVE path to the SHA1 HASH file, containing name and family of known malware
<i>sets</i>	The ids of the SubSets considered by the classifier
<i>percent</i>	Percent of all available examples used for training

MALWARE DETECTOR method's definitions

5.3.2.2 `MalwareDetector::~~MalwareDetector () [inline]`

Destructor.

Destructor of the inherited class MalwareClassifier

5.3.3 Member Function Documentation

5.3.3.1 `void MalwareDetector::evaluate (const uint nb, const uint to, const uint from) const [private], [virtual]`

Evaluate.

Classifies nb examples in the domain selected and show results

Parameters

<i>nb</i>	Number of examples to classify
<i>to</i>	Position of the first file of the domain
<i>from</i>	Position of the last file of the domain

(0 <= nb <= from - to)

Implements [NaiveBayesClassifier](#).

5.3.3.2 void `MalwareDetector::train` () [private],[virtual]

Train.

Train our classifier with m_sizeTrain examples

Implements [NaiveBayesClassifier](#).

The documentation for this class was generated from the following files:

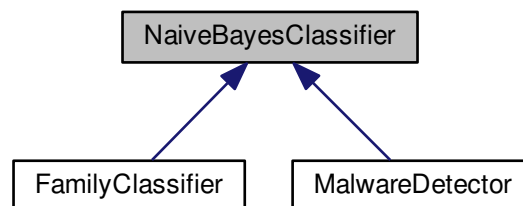
- [Classifiers.hpp](#)
- [Classifiers.cpp](#)

5.4 NaiveBayesClassifier Class Reference

Parent Classifier.

```
#include <Classifiers.hpp>
```

Inheritance diagram for NaiveBayesClassifier:



Public Member Functions

- [NaiveBayesClassifier](#) (const boost::filesystem::path &pDS, const boost::filesystem::path &pHD, const std::set< int > &sets, const uint percent)
Constructor.
- virtual [~NaiveBayesClassifier](#) ()
Destructor.
- void [evaluate](#) (const uint nbTests) const
Classify.

Protected Member Functions

- virtual void [train](#) ()=0
Train.
- virtual void [evaluate](#) (const uint nb, const uint to, const uint from) const =0
Evaluate.
- virtual const std::string [getClass](#) (const boost::filesystem::path &filePath) const
getClass

Protected Attributes

- const uint [m_percent](#)
Percent of all availbale examples used for training.
- uint [m_total](#)
Total of all the examples available.
- uint [m_sizeTrain](#)
*Maximum number of examples to train our classifier (= total * percent)*
- uint [m_sizeTest](#)
Maximum number of examples to test our classifier (= total - train)
- const boost::filesystem::path [m_pathHashDico](#)
RELATIVE path to the SHA1 HASH file, containing name and family of known malware.
- const boost::filesystem::path [m_pathDataSet](#)
RELATIVE path to the directory containing all the examples.
- std::set< std::string > [m_hashMalware](#)
All the name of known malware (read from m_pathHashDico)
- std::vector< [Class](#) * > [m_class](#)
Contains each class the classifier study.
- const std::set< int > [m_sets](#)
Contains the id of the SubSets considered by the classifier.

5.4.1 Detailed Description

Parent Classifier.

This class is used to create a classifier

5.4.2 Constructor & Destructor Documentation

5.4.2.1 [NaiveBayesClassifier::NaiveBayesClassifier](#) (const boost::filesystem::path & *pDS*, const boost::filesystem::path & *pHD*, const std::set< int > & *sets*, const uint *percent*)

Constructor.

Constructor of the parent class [NaiveBayesClassifier](#)

Parameters

<i>pDS</i>	RELATIVE path to the directory containing all the examples
<i>pHD</i>	RELATIVE path to the SHA1 HASH file, containing name and family of known malware
<i>sets</i>	The ids of the sets considered by the classifier
<i>percent</i>	Percent of all availbale examples used for training

NAIVE BAYES CLASSIFIER method's definitions

5.4.2.2 `virtual NaiveBayesClassifier::~~NaiveBayesClassifier () [inline],[virtual]`

Destructor.

Vitrual Destructor of the parent class [NaiveBayesClassifier](#)

5.4.3 Member Function Documentation

5.4.3.1 `void NaiveBayesClassifier::evaluate (const uint nbTests) const`

Classify.

Classifies examples in the know and/or unkwon domain

Parameters

<i>nbTests</i>	number of examples to classify
----------------	--------------------------------

5.4.3.2 `virtual void NaiveBayesClassifier::evaluate (const uint nb, const uint to, const uint from) const [protected],
[pure virtual]`

Evaluate.

Classifies nb examples in the domain selected and show results

Parameters

<i>nb</i>	Number of examples to classify
<i>to</i>	Position of the first file of the domain
<i>from</i>	Position of the last file of the domain

(0 <= nb <= from - to)

Implemented in [MalwareDetector](#), and [FamilyClassifier](#).

5.4.3.3 `const string NaiveBayesClassifier::getClass (const boost::filesystem::path & filePath) const [protected],
[virtual]`

getClass

Returns the most likely class of the example

Parameters

<i>filePath</i>	RELATIVE path to the example
-----------------	------------------------------

Returns

The most likely class

5.4.3.4 `virtual void NaiveBayesClassifier::train ()` [protected], [pure virtual]

Train.

Train our classifier with `m_sizeTrain` examples

Implemented in [MalwareDetector](#), and [FamilyClassifier](#).

5.4.4 Member Data Documentation

5.4.4.1 `std::vector<Class*> NaiveBayesClassifier::m_class` [protected]

Contains each class the classifier study.

5.4.4.2 `std::set<std::string> NaiveBayesClassifier::m_hashMalware` [protected]

All the name of known malware (read from `m_pathHashDico`)

5.4.4.3 `const boost::filesystem::path NaiveBayesClassifier::m_pathDataSet` [protected]

RELATIVE path to the directory containing all the examples.

5.4.4.4 `const boost::filesystem::path NaiveBayesClassifier::m_pathHashDico` [protected]

RELATIVE path to the SHA1 HASH file, containing name and family of known malware.

5.4.4.5 `const uint NaiveBayesClassifier::m_percent` [protected]

Percent of all available examples used for training.

5.4.4.6 `const std::set<int> NaiveBayesClassifier::m_sets` [protected]

Contains the id of the SubSets considered by the classifier.

5.4.4.7 `uint NaiveBayesClassifier::m_sizeTest` [protected]

Maximum number of examples to test our classifier (= total - train)

5.4.4.8 uint NaiveBayesClassifier::m_sizeTrain [protected]

Maximum number of examples to train our classifier (= total * percent)

5.4.4.9 uint NaiveBayesClassifier::m_total [protected]

Total of all the examples available.

The documentation for this class was generated from the following files:

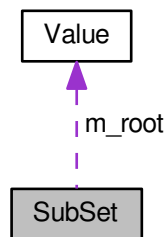
- [Classifiers.hpp](#)
- [Classifiers.cpp](#)

5.5 SubSet Class Reference

class [SubSet](#)

```
#include <VectorSpace.hpp>
```

Collaboration diagram for SubSet:



Public Member Functions

- [SubSet](#) (const int id)
Constructor.
- [~SubSet](#) ()
Destructor.
- void [add](#) (const std::string &value)
Add a value to the graph.
- const int [getId](#) () const
Get id.
- const std::string & [name](#) () const
Get name.
- const uint [getOcc](#) (const std::string &value)
Get occurrence.
- const uint [getOcc](#) () const
Get id.
- const long double [getLogProbability](#) (const std::string &value)
Get natural logarithmic probability.

Static Public Member Functions

- static const std::string & [name](#) (const uint id)
Get name.

Private Attributes

- const int [m_id](#)
Id of the [SubSet](#).
- uint [m_nbValue](#)
Number of values in the [SubSet](#).
- long double [m_nbLValue](#)
Natural logarithmic number of value.
- Value [m_root](#)
First node of the graph of Values.

Static Private Attributes

- static const std::vector< std::string > [setNames](#)
Stored the name of an id corresponding to a [SubSet](#).

Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [SubSet](#) &ss)
Operator <<.

5.5.1 Detailed Description

class [SubSet](#)

This class represents a set Contains the id of the set and a graph with all the value read in the examples which belong to this set

5.5.2 Constructor & Destructor Documentation

5.5.2.1 [SubSet::SubSet \(const int *id* \)](#) `[inline]`

Constructor.

Only constructor of the class [SubSet](#)

5.5.2.2 [SubSet::~~SubSet \(\)](#) `[inline]`

Destructor.

Destructor of the class [SubSet](#)

5.5.3 Member Function Documentation

5.5.3.1 [void SubSet::add \(const std::string & *value* \)](#)

Add a value to the graph.

Adds a new value to the graph which start at [m_root](#)

Parameters

<i>value</i>	The value to add
--------------	------------------

SUBSET method's definitions

5.5.3.2 `const int SubSet::getId () const` `[inline]`

Get id.

Returns the id of the [SubSet](#)

Returns

The id of the [SubSet](#)

5.5.3.3 `const long double SubSet::getLogProbability (const std::string & value)` `[inline]`

Get natural logarithmic probability.

Returns the natural logarithmic propobability of occurence of a value in the [SubSet](#)

Parameters

<i>value</i>	The value to check the probability
--------------	------------------------------------

Returns

The natural logarithmic propobability of occurence of a value

5.5.3.4 `const uint SubSet::getOcc (const std::string & value)`

Get occurence.

Returns the occurence of specified value in the [SubSet](#)

Parameters

<i>value</i>	The value to return the occurence
--------------	-----------------------------------

Returns

The occurence of the value

5.5.3.5 `const uint SubSet::getOcc () const` `[inline]`

Get id.

Returns the sum of occurrence of all the values in the [SubSet](#)

Returns

The sum of occurrence

5.5.3.6 `const std::string& SubSet::name () const` `[inline]`

Get name.

Returns the name of the [SubSet](#)

Returns

The name of the [SubSet](#)

5.5.3.7 `static const std::string& SubSet::name (const uint id)` `[inline],[static]`

Get name.

Returns the name of a [SubSet](#)

Parameters

<i>id</i>	The set's id to return the name
-----------	---------------------------------

Returns

The name of a [SubSet](#)

5.5.4 Friends And Related Function Documentation

5.5.4.1 `std::ostream& operator<< (std::ostream & out, const SubSet & ss)` `[friend]`

Operator <<.

Overloads the << operator to print a [Value](#)

5.5.5 Member Data Documentation

5.5.5.1 `const int SubSet::m_id` `[private]`

Id of the [SubSet](#).

5.5.5.2 `long double SubSet::m_nbLValue` [private]

Natural logarithmic number of value.

5.5.5.3 `uint SubSet::m_nbValue` [private]

Number of values in the [SubSet](#).

5.5.5.4 `Value SubSet::m_root` [private]

First node of the graph of Values.

5.5.5.5 `const vector< string > SubSet::setNames` [static], [private]

Initial value:

```
=  
{ "Hardware components",  
  "Requested permissions",  
  "App components",  
  "Filtered intents",  
  "Restricted API calls",  
  "Used permissions",  
  "Suspicious API calls",  
  "Network addresses" }
```

Stored the name of an id corresponding to a [SubSet](#).

The documentation for this class was generated from the following files:

- [VectorSpace.hpp](#)
- [VectorSpace.cpp](#)

5.6 Value Class Reference

class [Value](#)

```
#include <VectorSpace.hpp>
```

Public Member Functions

- [Value](#) ()
Constructor.
- [Value](#) (const std::string &value)
Constructor.
- [~Value](#) ()
Destructor.
- void [addOcc](#) ()
Add occurence.
- const uint [getOcc](#) () const
Get occurence.
- void [addNext](#) (const std::string &value)
Add a value to the graph.
- [Value](#) * [goNext](#) (const std::string &value)
Go to next.
- const std::string & [getValue](#) () const
Get value.
- const bool [isInNext](#) (const std::string &value) const
Check if the value exists in the successors.

Private Member Functions

- void [print](#) (std::ostream &out, const std::string offset) const
Print the value.

Private Attributes

- uint [m_occ](#)
Number of times the value appear in the [SubSet](#).
- const std::string [m_value](#)
String stored in the [Value](#).
- std::vector< [Value](#) * > [m_next](#)
Adresses of all the following Values.

Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [Value](#) &v)
Operator <<.

5.6.1 Detailed Description

class [Value](#)

This class is used to represents the node of a graph

5.6.2 Constructor & Destructor Documentation

5.6.2.1 Value::Value () [inline]

Constructor.

Default constructor of the class [Value](#)

5.6.2.2 Value::Value (const std::string & value) [inline]

Constructor.

Constructor of the class [Value](#)

Parameters

<i>value</i>	value contained by the node
--------------	-----------------------------

5.6.2.3 Value::~~Value () [inline]

Destructor.

Destructor of the class [Value](#) Delete recursively all the [Value](#) of the [SubSet](#)

5.6.3 Member Function Documentation

5.6.3.1 void Value::addNext (const std::string & value) [inline]

Add a value to the graph.

Adds a new value to the successors of the current node

Parameters

<i>value</i>	The value to add
--------------	------------------

5.6.3.2 void Value::addOcc () [inline]

Add occurrence.

Adds one to the number of value's occurrence

5.6.3.3 const uint Value::getOcc () const [inline]

Get occurrence.

Returns the number of times the value appear in the [SubSet](#)

Returns

true if added, false if not

5.6.3.4 `const std::string& Value::getValue () const` `[inline]`

Get value.

Returns the value of the current node

Returns

value

5.6.3.5 `Value * Value::goNext (const std::string & value)` `[inline]`

Go to next.

Returns the adress of the next [Value](#) containing value

Parameters

<i>value</i>	The value to look for in the nexts
--------------	------------------------------------

Returns

Adress of the next [Value](#), NULL if it doesn't exist

5.6.3.6 `const bool Value::isInNext (const std::string & value) const` `[inline]`

Check if the value exists in the successors.

Checks if a value is in the successors of the current node

Parameters

<i>value</i>	The value to check
--------------	--------------------

Returns

true if yes, false if not

VALUE method's definitions

5.6.3.7 `void Value::print (std::ostream & out, const std::string offset) const` `[private]`

Print the value.

Prints recursively the value and all the nexts

5.6.4 Friends And Related Function Documentation

5.6.4.1 `std::ostream& operator<< (std::ostream & out, const Value & v)` `[friend]`

Operator <<.

Overloads the << operator to print a [Value](#)

5.6.5 Member Data Documentation

5.6.5.1 `std::vector<Value*> Value::m_next` `[private]`

Adresses of all the following Values.

5.6.5.2 `uint Value::m_occ` `[private]`

Number of times the value appear in the [SubSet](#).

5.6.5.3 `const std::string Value::m_value` `[private]`

String stored in the [Value](#).

The documentation for this class was generated from the following files:

- [VectorSpace.hpp](#)
- [VectorSpace.cpp](#)

Chapter 6

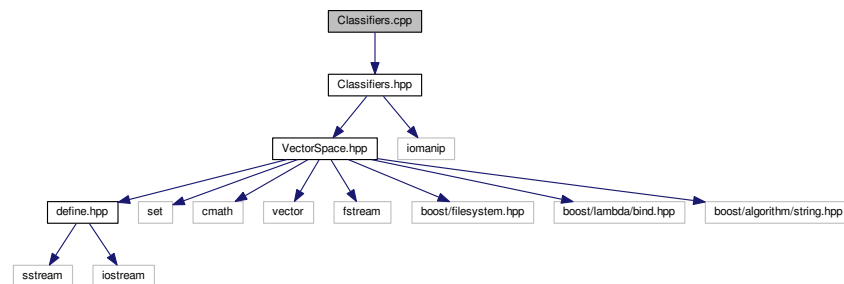
File Documentation

6.1 Classifiers.cpp File Reference

Contains the implemented methods of the different classifiers.

```
#include "Classifiers.hpp"
```

Include dependency graph for Classifiers.cpp:



6.1.1 Detailed Description

Contains the implemented methods of the different classifiers.

Author

Aurelien Bec

Date

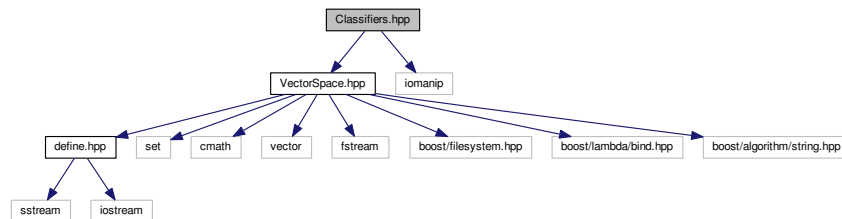
November 17th 2017

6.2 Classifiers.hpp File Reference

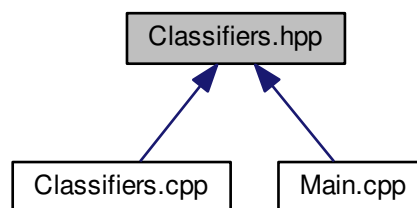
Contains the definition of different classifiers.

```
#include "VectorSpace.hpp"
#include <iomanip>
```

Include dependency graph for Classifiers.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [NaiveBayesClassifier](#)
Parent Classifier.
- class [FamilyClassifier](#)
Malware family classifier (inherited from [NaiveBayesClassifier](#))
- class [MalwareDetector](#)
Malware detector (inherited from [NaiveBayesClassifier](#))

6.2.1 Detailed Description

Contains the definition of different classifiers.

Author

Aurelien Bec

Date

November 17th 2017

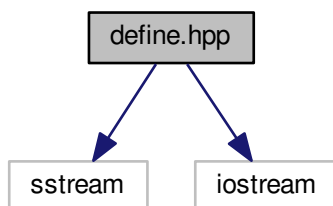
6.3 define.hpp File Reference

Contains the global definitions for the project.

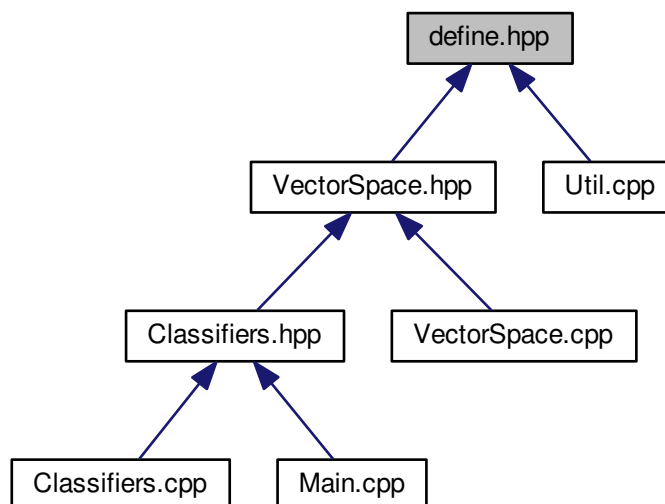
```
#include <sstream>
```

```
#include <iostream>
```

Include dependency graph for define.hpp:



This graph shows which files directly or indirectly include this file:



Macros

- #define `DEBUG_ON` false
- #define `BAD_BOUND` 30
- #define `GOOD_BOUND` 20
- #define `DEFAULT_D_PERCENT` 66

- `#define DEFAULT_D_NBTESTS 1000`
- `#define DEFAULT_C_PERCENT 66`
- `#define DEFAULT_C_NBTESTS 2000`
- `#define DEFAULT_MINNUMBERSAMPLE 20`
- `#define DEFAULT_SETS {1, 2, 3, 4, 5, 6, 7, 8}`
- `#define DEFAULT_PATH_HASHDICO "drebin/sha256_family.csv"`
- `#define DEFAULT_PATH_DATASET "drebin/feature_vectors"`
- `#define TOKENIZERS "\t ;, . ! () ? - > / % = _ "`

Functions

- `const std::string stringNULL ("0")`
- `const std::string colorizedValue (const double value, const bool invert)`
- `const std::string toLower (const std::string &in)`
- `const long double divide (const long double a, const long double b)`

6.3.1 Detailed Description

Contains the global definitions for the project.

Author

Aurelien Bec

Date

November 17th 2017

6.3.2 Macro Definition Documentation

6.3.2.1 `#define BAD_BOUND 30`

6.3.2.2 `#define DEBUG_ON false`

6.3.2.3 `#define DEFAULT_C_NBTESTS 2000`

6.3.2.4 `#define DEFAULT_C_PERCENT 66`

6.3.2.5 `#define DEFAULT_D_NBTESTS 1000`

6.3.2.6 `#define DEFAULT_D_PERCENT 66`

6.3.2.7 `#define DEFAULT_MINNUMBERSAMPLE 20`

6.3.2.8 `#define DEFAULT_PATH_DATASET "drebin/feature_vectors"`

6.3.2.9 `#define DEFAULT_PATH_HASHDICO "drebin/sha256_family.csv"`

6.3.2.10 `#define DEFAULT_SETS {1, 2, 3, 4, 5, 6, 7, 8}`

6.3.2.11 `#define GOOD_BOUND 20`

6.3.2.12 `#define TOKENIZERS "t :;,!()?->/%=_ "`

6.3.3 Function Documentation

6.3.3.1 `const std::string coloredValue (const double value, const bool invert)`

6.3.3.2 `const long double divide (const long double a, const long double b)`

6.3.3.3 `const std::string stringNULL ("0")`

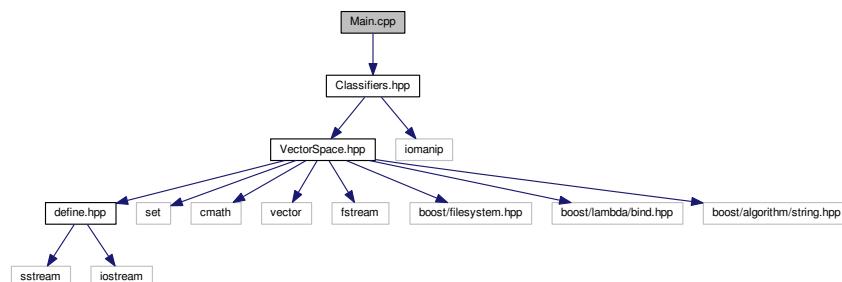
6.3.3.4 `const std::string toLower (const std::string & in)`

6.4 Main.cpp File Reference

Contains the main function.

```
#include "Classifiers.hpp"
```

Include dependency graph for Main.cpp:



Functions

- `int main (int argc, char *argv[])`

6.4.1 Detailed Description

Contains the main function.

Author

Aurelien Bec

Date

November 17th 2017

6.4.2 Function Documentation

6.4.2.1 `int main (int argc, char * argv[])`

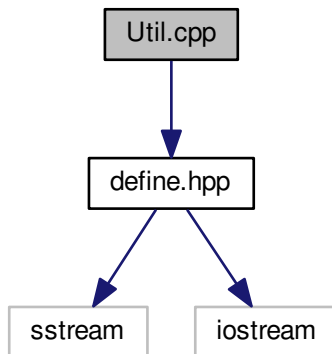
6.5 README.md File Reference

6.6 Util.cpp File Reference

Contains some useful functions for the project.

```
#include "define.hpp"
```

Include dependency graph for Util.cpp:



Functions

- `const string toLower (const string &in)`
- `const string coloredValue (const double value, const bool invert)`
- `const long double divide (const long double a, const long double b)`

6.6.1 Detailed Description

Contains some useful functions for the project.

Author

Aurelien Bec

Date

November 17th 2017

6.6.2 Function Documentation

6.6.2.1 `const string coloredValue (const double value, const bool invert)`

6.6.2.2 `const long double divide (const long double a, const long double b)`

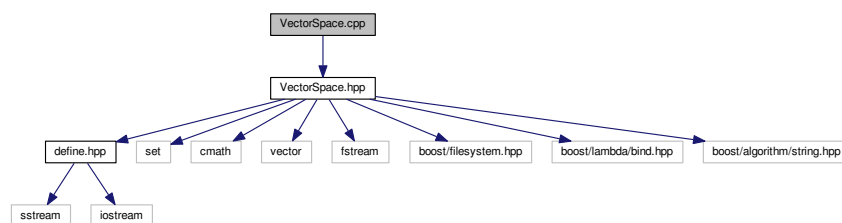
6.6.2.3 `const string toLower (const string & in)`

6.7 VectorSpace.cpp File Reference

Contains the implemented methods of VectorSpace and the other class need.

```
#include "VectorSpace.hpp"
```

Include dependency graph for VectorSpace.cpp:



Functions

- ostream & `operator<<` (ostream &out, const [Class](#) &c)
- ostream & `operator<<` (ostream &out, const [SubSet](#) &ss)
- ostream & `operator<<` (ostream &out, const [Value](#) &v)

6.7.1 Detailed Description

Contains the implemented methods of VectorSpace and the other class need.

Author

Aurelien Bec

Date

November 17th 2017

6.7.2 Function Documentation

6.7.2.1 `ostream& operator<< (ostream & out, const Class & c)`

printing method's definitions

6.7.2.2 ostream& operator<< (ostream & out, const SubSet & ss)

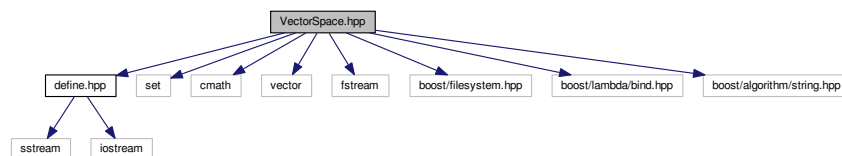
6.7.2.3 ostream& operator<< (ostream & out, const Value & v)

6.8 VectorSpace.hpp File Reference

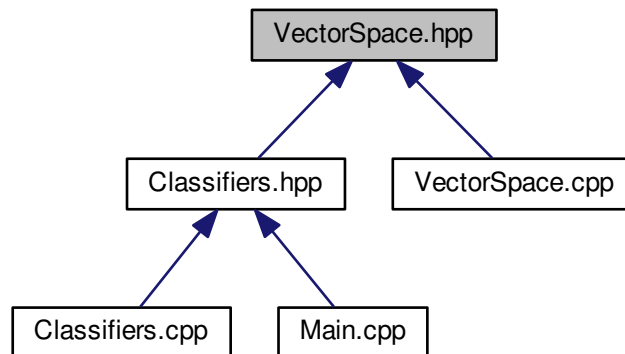
Contains the definition of VectorSpace and the other class need.

```
#include "define.hpp"
#include <set>
#include <cmath>
#include <vector>
#include <fstream>
#include <boost/filesystem.hpp>
#include <boost/lambda/bind.hpp>
#include <boost/algorithm/string.hpp>
```

Include dependency graph for VectorSpace.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [Value](#)
class *Value*
- class [SubSet](#)
class *SubSet*
- class [Class](#)
Parent Classifier.

6.8.1 Detailed Description

Contains the definition of VectorSpace and the other class need.

Author

Aurelien Bec

Date

November 17th 2017

Index

- ~Class
 - Class, [10](#)
- ~FamilyClassifier
 - FamilyClassifier, [15](#)
- ~MalwareDetector
 - MalwareDetector, [17](#)
- ~NaiveBayesClassifier
 - NaiveBayesClassifier, [20](#)
- ~SubSet
 - SubSet, [23](#)
- ~Value
 - Value, [28](#)

- add
 - SubSet, [23](#)
- addNext
 - Value, [28](#)
- addOcc
 - Value, [28](#)

- BAD_BOUND
 - define.hpp, [34](#)

- Class, [9](#)
 - ~Class, [10](#)
 - Class, [10](#)
 - getLogProbability, [11](#)
 - getSetId, [11](#)
 - m_name, [13](#)
 - m_nbMan, [13](#)
 - m_nbValue, [13](#)
 - m_pL, [13](#)
 - m_sets, [13](#)
 - name, [11](#)
 - nbManifests, [12](#)
 - operator<<, [12](#)
 - setNbValue, [12](#)
 - setProbability, [12](#)
 - study, [12](#)

- Classifiers.cpp, [31](#)

- Classifiers.hpp, [32](#)

- colorizedValue
 - define.hpp, [35](#)
 - Util.cpp, [37](#)

- DEBUG_ON
 - define.hpp, [34](#)

- DEFAULT_C_NBTESTS
 - define.hpp, [34](#)

- DEFAULT_C_PERCENT

- define.hpp, [34](#)

- DEFAULT_D_NBTESTS

- define.hpp, [34](#)

- DEFAULT_D_PERCENT

- define.hpp, [34](#)

- DEFAULT_MINNUMBERSAMPLE

- define.hpp, [34](#)

- DEFAULT_PATH_DATASET

- define.hpp, [34](#)

- DEFAULT_PATH_HASHDICO

- define.hpp, [34](#)

- DEFAULT_SETS

- define.hpp, [34](#)

- define.hpp, [33](#)

- BAD_BOUND, [34](#)

- colorizedValue, [35](#)

- DEBUG_ON, [34](#)

- DEFAULT_C_NBTESTS, [34](#)

- DEFAULT_C_PERCENT, [34](#)

- DEFAULT_D_NBTESTS, [34](#)

- DEFAULT_D_PERCENT, [34](#)

- DEFAULT_MINNUMBERSAMPLE, [34](#)

- DEFAULT_PATH_DATASET, [34](#)

- DEFAULT_PATH_HASHDICO, [34](#)

- DEFAULT_SETS, [34](#)

- divide, [35](#)

- GOOD_BOUND, [35](#)

- stringNULL, [35](#)

- TOKENIZERS, [35](#)

- toLower, [35](#)

- divide

- define.hpp, [35](#)

- Util.cpp, [37](#)

- evaluate

- FamilyClassifier, [15](#)

- MalwareDetector, [17](#)

- NaiveBayesClassifier, [20](#)

- FamilyClassifier, [13](#)

- ~FamilyClassifier, [15](#)

- evaluate, [15](#)

- FamilyClassifier, [14](#)

- train, [15](#)

- GOOD_BOUND

- define.hpp, [35](#)

- getClass

- NaiveBayesClassifier, [20](#)

- getId

- SubSet, 24
- getLogProbability
 - Class, 11
 - SubSet, 24
- getOcc
 - SubSet, 24
 - Value, 28
- getSetId
 - Class, 11
- getValue
 - Value, 29
- goNext
 - Value, 29
- isInNext
 - Value, 29
- m_class
 - NaiveBayesClassifier, 21
- m_hashMalware
 - NaiveBayesClassifier, 21
- m_id
 - SubSet, 25
- m_name
 - Class, 13
- m_nbLValue
 - SubSet, 25
- m_nbMan
 - Class, 13
- m_nbValue
 - Class, 13
 - SubSet, 26
- m_next
 - Value, 30
- m_occ
 - Value, 30
- m_pathDataSet
 - NaiveBayesClassifier, 21
- m_pathHashDico
 - NaiveBayesClassifier, 21
- m_percent
 - NaiveBayesClassifier, 21
- m_pL
 - Class, 13
- m_root
 - SubSet, 26
- m_sets
 - Class, 13
 - NaiveBayesClassifier, 21
- m_sizeTest
 - NaiveBayesClassifier, 21
- m_sizeTrain
 - NaiveBayesClassifier, 21
- m_total
 - NaiveBayesClassifier, 22
- m_value
 - Value, 30
- main
 - Main.cpp, 36
- Main.cpp, 35
 - main, 36
- MalwareDetector, 16
 - ~MalwareDetector, 17
 - evaluate, 17
 - MalwareDetector, 17
 - train, 18
- NaiveBayesClassifier, 18
 - ~NaiveBayesClassifier, 20
 - evaluate, 20
 - getClass, 20
 - m_class, 21
 - m_hashMalware, 21
 - m_pathDataSet, 21
 - m_pathHashDico, 21
 - m_percent, 21
 - m_sets, 21
 - m_sizeTest, 21
 - m_sizeTrain, 21
 - m_total, 22
 - NaiveBayesClassifier, 19
 - train, 21
- name
 - Class, 11
 - SubSet, 25
- nbManifests
 - Class, 12
- operator<<
 - Class, 12
 - SubSet, 25
 - Value, 30
 - VectorSpace.cpp, 37, 38
- print
 - Value, 29
- README.md, 36
- setNames
 - SubSet, 26
- setNbValue
 - Class, 12
- setProbability
 - Class, 12
- stringNULL
 - define.hpp, 35
- study
 - Class, 12
- SubSet, 22
 - ~SubSet, 23
 - add, 23
 - getId, 24
 - getLogProbability, 24
 - getOcc, 24
 - m_id, 25
 - m_nbLValue, 25
 - m_nbValue, 26

- m_root, [26](#)
- name, [25](#)
- operator<<, [25](#)
- setNames, [26](#)
- SubSet, [23](#)

TOKENIZERS

- define.hpp, [35](#)

toLower

- define.hpp, [35](#)
- Util.cpp, [37](#)

train

- FamilyClassifier, [15](#)
- MalwareDetector, [18](#)
- NaiveBayesClassifier, [21](#)

Util.cpp, [36](#)

- colorizedValue, [37](#)
- divide, [37](#)
- toLower, [37](#)

Value, [26](#)

- ~Value, [28](#)
- addNext, [28](#)
- addOcc, [28](#)
- getOcc, [28](#)
- getValue, [29](#)
- goNext, [29](#)
- isInNext, [29](#)
- m_next, [30](#)
- m_occ, [30](#)
- m_value, [30](#)
- operator<<, [30](#)
- print, [29](#)
- Value, [28](#)

VectorSpace.cpp, [37](#)

- operator<<, [37](#), [38](#)

VectorSpace.hpp, [38](#)